LINE SEARCH MULTILEVEL OPTIMIZATION AS COMPUTATIONAL METHODS FOR DENSE OPTICAL FLOW

EL MOSTAFA KALMOUN, LUIS GARRIDO, AND VICENT CASELLES

Abstract. We evaluate the performance of different optimization techniques developed in the context of optical flow computation with different variational models. In particular, based on truncated Newton methods (TN) that have been an effective approach for large-scale unconstrained optimization, we develop the use of efficient multilevel schemes for computing the optical flow. More precisely, we evaluate the performance of a standard unidirectional multilevel algorithm - called multiresolution optimization (MR/OPT), to a bidrectional multilevel algorithm - called full multigrid optimization (FMG/OPT). The FMG/OPT algorithm treats the coarse grid correction as an optimization search direction and eventually scales it using a line search. Experimental results on different image sequences using four models of optical flow computation show that the FMG/OPT algorithm outperforms both the TN and MR/OPT algorithms in terms of the computational work and the quality of the optical flow estimation.

1. Introduction

The problem of optical flow computation consists in finding the 2D-displacement field that represents apparent motion of objects in a sequence of images. Many efforts have been devoted to it in computer vision and applied mathematics [25, 35, 18, 3, 4, 11, 40, 44, 12, 10, 31, 27, 1, 2, 6, 32, 33, 36]. The computation of optical flow is usually based on the conservation of some property during motion, either the object's gray level or their shape properties. In a variational setting, the problem is usually formulated as a minimization of an energy function, which is a weighted sum of two terms: a data term coming from the motion modeling and a smoothing term as a result of the regularization process. For standard optical flow algorithms, the data term is usually based on a brightness constancy assumption, which assumes that the object illumination does not

Key words and phrases. Optical flow, line search multigrid optimization, multiresolution, truncated Newton.

This research was initiated while the first author was visiting Centre de Recerca Matemàtica and Universitat Pompeu Fabra in Barcelona during 2008. The first author was partially supported by FRGS grant UUM/RIMC/P-30 (S/O code 11872) offered by Ministry of. Higher Education Malaysia. The second and third authors acknowledge partial support by MICINN project, reference MTM2009-08171, and by GRC reference 2009 SGR 773. The third author also acknowledges partial support by "ICREA Acadèmia" prize for excellence in research funded both by the Generalitat de Catalunya.

change along its motion trajectory. The regularization process ensures that the optical flow estimation problem is well-posed. Many regularization terms have been investigated in the last two decades ranging from isotropic to anisotropic. Isotropic smoothers tend to blur motion edges while the anisotropic ones require additional computational resources.

Two strategies might be conducted for the numerical minimization of an energy functional arising in a variational framework like above. The first one which is called minimize-discretize is achieved by discretizing and solving the corresponding Euler-Lagrange equations. In the second approach called discretize-minimize one can use directly numerical optimization methods for solving a discrete version of the problem. While the first approach has been commonly used for variational optical flow computation, the second computational strategy has been less investigated in this context, to the best of our knowledge.

Adopting the first strategy, efficient algorithms have been designed to approximate the optical flow. In particular, in [1, 2] the authors used a multiscale approximation to solve the corresponding Euler-Lagrange equations using the Nagel-Enkelmann regularization term [36]. Thus, they computed a series of approximations where each one solves a regularized version of the Euler-Lagrange equations starting from the previous one while keeping the original grid fixed. A different approach is proposed in [9, 41] where the authors use fixed point iterations to solve the corresponding Euler-Lagrange equations, fully implicit in the smoothness term and semi-implicit in the data term. Still, this fixed point iteration leads to implicit equations and they linearize some of their terms using a warping technique. The equations obtained are fully linearized using a lagged diffusivity method [9, 41, 10]. The final linear system is solved using a linear solver like a Gauss-Seidel type method, or a SOR method. The connections with warping are detailed in [9, 41].

On the other hand, in order to develop efficient and accurate algorithms working in real time, there have been recent efforts to improve performance of optical flow algorithms using multilevel techniques. We distinguish at this stage between two classes of multilevel algorithms. The first one known as a coarse-to-fine multiresolution uses a sequence of coarse grid subproblems to find a good initialization for the finest grid problem that avoids possible local minima. We shall refer to this strategy in this paper by multiresolution. The second strategy alternates between solution relaxations using the underlying algorithm and solution corrections obtained from a sequence of sub-problems defined on coarse grids. This leads to recursive algorithms like the so-called V- or W-cycle, which traverse between fine and coarse grids in the mesh hierarchy. We will reserve the term multigrid for it. In the case of elliptic PDEs - and for a wide class of problems, multigrid methods are known to outperform multiresolution methods.

However, being straightforward to implement, multiresolution methods were more used in computer vision and particularly for motion estimation; e.g. [18, 3, 42, 40, 44, 15]. For instance, Enkelmann [18] developed a coarse-to-fine algorithm for oriented smoothed optical flow. Also, Cohen and Herlin [15] used recently a multiresolution technique with nonuniform sampling for total variation regularization.

For the seek of optimal performance, multigrid schemes were developped to solve the resulting Euler-Lagrange equations for both isotropic and anisotropic regularizations, see [21, 45, 20, 27, 26, 12, 10] and references therein. The first attempts are due to Glazer [21] and Terzopoulos [45] using standard multigrid components. Improvement in performance were reported on simple synthetic images and later on [42] standard multigrids were stated to be not appropriate due to a possible information conflict between different scales. However, the method was recently better adapted for optical flow computation by tuning the multigrid components. In [20], an entirely algebraic multigrid approach was developed for a weighted anisotropic regularization based model. A geometrical multigrid based on Galerkin coarse grid discretization approximation was developped for the classical Horn-Schunck method in [27]. This algorithm was extended and paralelized in [26] for real time computation of vector motion fields for 3D images. Another recent geometric multigrid investigation was presented in [12, 10] with anisotropic regularization (a full account of it can be found in [10]).

All these works have considered data terms based on the brightness constancy assumption, which lead to less accurate optical flow fields when the image sequence contains illumination variations in the temporal domain, which may be often found in real images. In [14], a model is proposed for illumination invariant optical flow computation, previously introduced in [17] in the context of image registration. The brightness constancy assumption is replaced by the assumption that the shapes of the image move along the sequence. In this context, the terms of the Euler-Lagrange equation corresponding to the data attachment term which contains derivatives of the unit normal vector fields are highly nonlinear. They will not produce systems of equations with a symmetric and positive semi-definite matrix (usually after linearisation), the basic systems to which the previous multigrid methods for optical flow have been applied.

For that reason we follow in this paper the second strategy of discretize-optimize that allows to handle such kind of variational problems. Instead of computing the Euler-Lagrange equations of the energy model, discretizing and solving them, our approach is based on the use of numerical optimization methods to solve the discrete version of the energy, be either based on gray level constancy or shape properties. This leads to the need of developing efficient algorithms for the numerical resolution of a large scale optimization problem. Therefore only large scale unconstrained optimization methods are relevant to solve this variational problem. One of them is the truncated Newton method [16]. It requires only the computation of function and gradient values and has then suitable storage requirements for large-scale problems. However, due to the intensive computations that are required to solve the energy minimization problem, only multilevel versions of the method (multiresolution and multigrid) are expected to provide suitable performance.

In our context, the bidirectional multigrid scheme should be adapted directly to an optimization problem. Current research is still ongoing in this direction for solving some variational problems lacking a governing PDE in different fields. Recently [38, 29], the multigrid strategy has been extended to optimization problems for truncated Newton methods. Motivated by variational problems lacking a governing PDE, the multigrid optimization was derived from the Full Approximation Storage (FAS) [7] for nonlinear PDEs but applied directly in an optimization setting. With regard to nonlinear multigrids, the MG/OPT algorithm includes two safeguards that guarantee convergence: Bounds on the coarse grid correction that introduce a constrained optimization subproblem and a line search that eventually scales the coarse grid correction treated as a search direction. In [29], it has been shown that the coarse grid subproblem is a first-order approximation of the fine-grid problem. This justifies somehow the introduction of the two safeguards. The first-order approximation suggests that the correction will be only reliable near the restricted approximation and it relates at the same time the MG/OPT to the steepest descent method. The latter connection indicates that the coarse grid correction will not be typically a well-scaled descent direction which, in turn, implies that a line search should be performed to adjust the scale of the multigrid search direction. This may not be necessary as the MG/OPT is near to convergence since in that case it will provide a Newton-like search direction for which a search step equal to 1 will be likely accepted. These connections to both steepest descent and Newton method suggest that the MG/OPT will perform well far and near the solution.

Our aim is to develop the MG/OPT method for the computation of optical flow, a problem which is of considerable computational resource requirements. To the best of our knowledge, the MG/OPT method is studied here for the first time for optical flow computation. Several components of the MG/OPT technique have been tuned for high efficiency and the algorithm is fully evaluated with respect to one-way multiresolution optimization. The proposed numerical strategy can be adapted to the minimization of other nonlinear energy functionals like the illumination invariant model proposed in [14] or depth estimation in stereo problems. Although they are an important motivation for the development of the present techniques they will not be considered in this paper.

The outline of our paper is as follows. In Section 2, we start off with a review of the variational formulation of the optical flow problem. In Section 3, we recall the basics of the truncated Newton method. In Section 4, we present multilevel algorithms applied to optimization problems. First, we discuss the coarse-to-fine multiresolution strategy. Then, after recalling the idea of multigrid for linear systems, we describe its application to optimization-based problems. In Section 5, we outline some of the implementation details, namely the calculation of the objective functional for each of the four models considered in this paper, of its Hessian computation and of the image derivatives. In Section 6, we report our experimental results by considering three classical sequences of synthetic images: the translating tree, the diverging tree and the Yosemite sequences. In Section 7, we conclude the paper and indicate future research directions.

2. Variational models for optical flow

Let us consider a sequence of gray level images $I(t, x, y)$, $t \in [0, T]$, $(x, y) \in Q$, where Q denotes the image domain, which we assume to be a rectangle in $\mathbb{R}^2$. We shall either consider the case where $t \in [0, T]$, and the case where the sequence is sampled at the times $t_j = j\Delta t$, $j = 0, \dots, K$.
Assuming that the gray level of a point does not change over time we may write the constraint

$$I(t, x(t), y(t)) = I(0, x, y),$$

where $(x(t), y(t))$ is the apparent trajectory of the point $(x(0), y(0)) = (x, y)$. Differentiating with respect to t and denoting $(u(t, x, y), v(t, x, y)) = (x'(t), y'(t))$ we obtain the optical flow constraint

$$I^t + uI^x + vI^y = 0. \tag{2.1}$$

The vector field $w(t, x, y) := (u(t, x, y), v(t, x, y))$ is called optic flow and I^t, I^x, I^y denote the partial derivatives of I with respect to t, x, y, respectively. Clearly, the single constraint (2.1) is not sufficient to uniquely compute the two components (u, v) of the optic flow (this is called the aperture problem) and only gives the component of the flow normal to the image gradient, i.e., to the level lines of the image. As it is usual, in order to recover a unique flow field a regularization constraint is added. For that, we assume that the optic flow varies smoothly in space, or better, that is piecewise smooth in Q. This can be achieved by including a smoothness term of the form

$$R(w) := \int_Q G(\nabla I, \nabla u, \nabla v)\, dxdy, \tag{2.2}$$

where $G\colon \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is a suitable function. The case $G = \|\nabla u\|^2 + \|\nabla v\|^2$ corresponds to the Horn-Schunk model [25], the case $G = \mathrm{trace}((\nabla w)^T D(\nabla I)\nabla w)$ corresponds to the Nagel-Enkelmann model [36], the case $G = \sqrt{\|\nabla u\|^2 + \|\nabla v\|^2}$ or $G = \|\nabla u\| + \|\nabla v\|$ correspond to total variation regularization models. For a full account of this and the associated taxonomy, we refer to [46, 24].

Both data attachment and regularization terms can be combined into a single energy functional

$$\int_Q (I^t + uI^x + vI^y)^2\, dxdy + \alpha \int_Q G(\nabla I, \nabla u, \nabla v)\, dxdy, \tag{2.3}$$

where $\alpha > 0$ is the regularization parameter weighting the relative importance of both terms.

In case of illumination changes, the gray level constancy assumption is violated, and may be substituted by the constancy of the gradient [5, 41] which can be expressed in differential form as

$$\begin{aligned}\langle w, \nabla I^x \rangle &= 0 \\ \langle w, \nabla I^y \rangle &= 0.\end{aligned} \tag{2.4}$$

Other cases include the constancy of the gradient direction [13] or the assumption that the shapes of the image (identified as the level lines) move along the sequence [14] (this assumption has been used in [17] in the context of image registration). Higher derivative models have been studied in [41].

The above models (2.3), (2.4) do not take into account that video sequences are sampled in time so that our data has the form $I_j(x, y) := I(t_j, x, y)$, $t_j = j\Delta t$, $j = 0, \dots, K$. Without loss of generality, let us assume that $\Delta t = 1$. In that case, the gray level constancy may be expressed as

$$I_{j-1}(x, y) = I_j(x + u_j(x, y), y + v_j(x, y)), \tag{2.5}$$

where $(u_j(x, y), v_j(x, y))$ is the optical flow between images I_{j-1} and I_j. As argued in [1, 2] the linearized gray level constancy constraint (2.1) may not be a good approximation in case of large displacements and the form (2.5) may be more appropriate [10].

A corresponding energy functional can be obtained by combining the non linearized form of the brightness constancy assumption and a regularization term. For convenience, we assume that we want to compute the optical flow between two images $I_1(x, y)$ and $I_2(x, y)$. We may write the energy

$$\int_Q \Psi\left((I_1(x, y) - I_2(x + u(x, y), y + v(x, y)))^2\right) dxdy + \alpha \int_Q G(\nabla I, \nabla u, \nabla v)\, dxdy, \tag{2.6}$$

where $\Psi \colon \mathbb{R} \to \mathbb{R}$ is an increasing smooth function, and $\alpha > 0$. Examples of function G have been given above (see [46, 24, 10] for an account of the many different possibilities).

Observe that the energy (2.6) is nonlinear and non convex. In order to develop the basic numerical optimization methods, the variational optical flow problem is set into the simplified minimization form:

$$\min_w f(w), \tag{2.7}$$

where $f(w) := D(w) + \alpha R(w)$. Here D denotes a given data term based on either (2.1) or (2.6) and R denotes a regularization term being either quadratic or the total variation. More precisely, we consider in this paper the case of $G = \|\nabla u\|^2 + \|\nabla v\|^2$ or $G = \|\nabla u\| + \|\nabla v\|$. We also consider $\Psi(s^2) = s^2$ for $|s| \leq \gamma$ and $\Psi(s^2) = \gamma^2$ for $|s| > \gamma$.

As discussed in the introduction, we adopt the strategy discretize-optimize. This means that we shall first discretize the objective functional and then solve a finite dimensional but large scale optimization problem. Therefore only large scale unconstrained optimization methods are relevant to solve this variational problem. One of these methods that requires only the computation of function and gradient values and has suitable storage requirements for large-scale problems is the truncated Newton method [16]. We note that other choices of the underlying optimization procedure - like the BFGS quasi Newton method, are possible. The truncated Newton algorithm is embedded in a multilevel strategy, both multiresolution and multigrid.

3. Truncated Newton optimization

As it is well known, Newton methods are based on the second order Taylor approximation of the objective function to build an iterative process for approaching a local minimum. The step s_k to move from the current point w_k to a new iterate w_{k+1} is chosen to be a minimum of the quadratic model of f given by

$$q_k(s) = f_k + g_k^T s + \frac{1}{2} s^T H_k s, \tag{3.1}$$

where $f_k = f(w_k)$, $g_k = g(w_k)$ and $H_k = H(w_k)$. The Newton step s_k is then obtained by solving the linear system

$$H_k s = -g_k. \tag{3.2}$$

For large-scale problems, solving exactly this linear system will be very expensive. Truncated Newton methods (TN) use rather an iterative method to find an approximate solution to (3.2) and truncate the iterates as soon as a required accuracy is reached or whenever (in case when the Hessian matrix H_k is not positive definite), a negative curvature is detected. One of the most well known iterative methods within TN ones is the Preconditioned Conjugate Gradient algorithm (PCG), see Algorithm 2, due to its efficiency and modest memory requirements. In this context, we will refer to the process of finding the step s_k as inner iterations, while the process of updating w_k using the computed s_k will be called outer iterations. Our discussion in the sequel depends on the use of the PCG method as an inner solver.

Depending on how the step solution s_k of the quadratic model (3.1) is exploited, two broad classes of algorithms are distinguished: line search methods and trust region methods. Line search methods scale the step s_k by a factor α_k that approximately minimizes f along the line that passes through w_k in the direction s_k. On the other hand, trust region methods restrict the search for s_k to some region $\mathcal{B}_k$ around w_k in which the algorithm "trusts" that the model function q_k behaves like the objective function f. This paper focuses on the former, the line search method, whereas the comparison between both methods (line search and trust region) will be the object of future work.

There are three components in the truncated Newton method related to speeding up the convergence of inner iterations that might have a great impact on its overall efficiency: a) the truncation criterion, b) the preconditioning strategy and c) how to handle the case when the Hessian matrix is not positive definite. Indeed, for the latter, one of the advantages of trust region over line search is that negative curvature directions can be properly exploited. In the PGC algorithm (Algorithm 2), the inner iterations are truncated as soon as a negative curvature direction is detected, that is, when $p_j^T H_k p_j$ is negative. In our case, we replace the negative curvature test with the equivalent descent direction test [48], see lines 8-11 of Algorithm 2. From our practical experience, the descent direction test has a better numerical behavior than the negative curvature test.

For the other two components a) and b), we used a scaled two-step limited memory BFGS [37] with a diagonal scaling for preconditioning the CG method, and we truncate the inner iterations when the following criterion is satisfied:

$$||r_j||_{M_k^{-1}} \leq \zeta_k ||r_0||_{M_k^{-1}},$$

where r_j is the PCG residual at inner iteration j, M_k is the preconditioning matrix and

$$\zeta_k = \max\left(0.5/(k+1), ||r_0||_{M_k^{-1}}\right) \tag{3.3}$$

which are both being provided at outer iteration k, and where

$$||r_k||_{M_k^{-1}} = \sqrt{r_k M_k^{-1} r_k}.$$

The matrix M_k^{-1} may be computed easily if M_k is updated using the BFGS method [39].

Line search TN methods ensure that the new TN step s_k provides a good descent by performing a line minimization along this direction and then the new outer iterate becomes:

$$w_{k+1} = w_k + \alpha_k s_k.$$

Exact line search is avoided due to expensive function evaluations and normally a sufficient function decrease is obtained by imposing the Wolf's conditions:

$$f_{k+1} \leq f_k + c_1 \alpha_k g_k^T s_k \tag{3.4}$$

$$g_{k+1}^T s_k \geq c_2\, g_k^T s_k, \tag{3.5}$$

where $0 < c_1 < c_2 < 1$. In order to obtain a maximum decrease with a minimum number of function evaluations, interpolating polynomials is usually employed. Here in TN a cubic interpolation was used [34].

The algorithm associated to the outer iterations of the line search TN is shown in Algorithm 1. The algorithm iteratively updates w_k by computing a search direction and performing a line search until a given tolerance is reached. In our case we set tolerances on the gradient, ϵ_g, to detect local minima, and on the function values and iterate values, ϵ_f and ϵ_x to detect convergence. Note also that the preconditionning matrix M_k is updated here, which can be easily done from previous values of w_k and $g(w_k)$, see [39].

Algorithm 1 Line Search Truncated Newton (outer iterations of TN)

```
1: Initialization $w_0$, $M_0 = Id$, $\epsilon_g, \epsilon_f, \epsilon_x$
2: for $k = 0$ to $max\_outer$ do
3:    $g_k = \nabla f(x_k)$
4:    if $||g_k|| < \epsilon_g$ then
5:       exit with solution $w_k$
6:    end if
7:    Compute $s_k$ by calling Algorithm 2.
8:    Perform a line search to scale the step $s_k$ by $\alpha_k$.
9:    $w_{k+1} = w_k + \alpha_k s_k$
10:   Update $M_{k+1}$ by the BFGS formula
11:   if $|f_{k+1} - f_k| < \epsilon_f$ or $||w_{k+1} - w_k|| < \epsilon_x$ then
12:      exit with solution $w_k$
13:   end if
14: end for
```

Algorithm 2 Preconditioned Conjugate Gradient (inner iterations of TN)

```
1: Initialization: $z_0 = 0$, $r_0 = -g_k$, $v_0 = M_k^{-1} r_0$, $p_0 = v_0$, $\epsilon = 10^{-10}$, $\zeta_k$(see (3.3))
2: for $j = 0$ to $max\_inner$ do
3:    // Singularity test
4:    if $\left(|r_j^T v_j| < \epsilon \text{ or } |p_j^T H_k p_j| < \epsilon\right)$ then
5:       exit with $s_k = z_j$ (for $j = 0$ take $s_k = -g_k$)
6:    end if
7:    $\alpha_j = r_j^T v_j / p_j^T H_k p_j \quad , \quad z_{j+1} = z_j + \alpha_j p_j$
8:    // Descent Direction Test replaces Negative Curvature test
9:    if $(g_k^T z_{j+1} \geq g_k^T z_j - \epsilon)$ then
10:      exit with $s_k = z_j$ (for $j = 0$ take $s_k = -g_k$)
11:   end if
12:   $r_{j+1} = r_j - \alpha_j H_k p_j \quad , \quad v_{j+1} = M_k^{-1} r_{j+1}$
13:   // Truncation test (note that $||r_k||_{M_k^{-1}} = r_{j+1}^T v_{j+1}$)
14:   if $(r_{j+1}^T v_{j+1} \leq \zeta_k g_0^T v_0)$ then
15:      exit with $s_k = z_{j+1}$
16:   end if
17:   $\beta_j = r_{j+1}^T (v_{j+1} - v_j) / r_j^T v_j \quad , \quad p_{j+1} = v_{j+1} + \beta_j p_j.$
18: end for
```

4. Multilevel methods

Multilevel methods use a set of subproblems defined on coarser levels of resolution in order to improve the convergence to the global optimum of (2.7) when compared to using only one level of resolution. Using multiple levels of resolution allows to deal, in the case of optical flow computation, with large displacements and enables the convergence to the global minimum since some local minima may disappear at sufficient coarse resolutions.

Let us denote with Ω_i the image domain at level i, where $i = 0 \ldots r$, where $i = 0$ corresponds to the finest level of resolution and $i = r$ to the coarsest one. Grid spacing at the coarser grid Ω_{i+1} is usually twice the spacing at the grid Ω_i. Two approaches are currently widely used in the computer vision field, namely the multiresolution and the multigrid methods. Both are explained below.

4.1. **Multiresolution methods.** Multiresolution methods have been applied successfully in many computer vision and image analysis problems where the problem can be expressed as a global optimization problem. Multiresolution methods use a series of coarse to fine resolution levels to obtain an estimate of the solution to the problem. An initial estimate is obtained at the coarsest level. In our case, this estimate may be obtained by applying the TN algorithm (see Algorithm 1). The estimate is then extended to the next level of resolution where it is refined. This process is repeated until the finest level is reached, where the final estimate is obtained, see Algorithm 3, where the MR (multiresolution) method is shown. This function is called with $MR(L-1, x_{L-1,0})$, where L is the number of resolution levels and $x_{L-1,0}$ is the initial estimate on the coarsest.

The advantage of coarse-to-fine multiresolution is that a good initial guess may be obtained for the finest grid problem by estimating a solution to the problem using the coarser grids. However, for linear systems it is currently known that one-way multilevel methods do not reach the optimal efficiency of standard bidirectional multigrid methods, which are detailed in the next section.

4.2. **Multigrid methods.** Multigrid methods were originally developed to solve elliptic PDEs and at present are known to be among the most powerful numerical methods for improving computational efficiency of a wide class of equations. For a classical reference on multigrids, we refer to [7, 8]. The main characteristic of multigrid algorithms is based on the observation that different frequencies are present in the error of the solution of the finest grid problem. Some algorithms, called *smoothers* (such as Gauss-Seidel), are known to efficiently reduce the high frequency components of the error on a grid (or, in other words, the components whose "wavelength" is comparable to the grid's mesh size). However, these algorithms have a small effect on the low frequency error components. This is the reason why the application of schemes like Gauss-Seidel to solve a problem, for a given grid, effectively reduces the error in the first iterations of the procedure

Algorithm 3 Multiresolution method

```
1: function MR($i, x_{i,0}$)
2:     $i = L - 1$: coarsest level of resolution
3:     $x_{i,0}$: initial estimate
4: repeat
5:    Apply TN with initial estimate $x_{i,0}$
6:    Let $x_{i,1}$ be the result
7:    if $i > 0$ then
8:       // Prolonge current estimate to next finer level
9:       $x_{i-1,0} = P x_{i,1}$
10:   end if
11:   $i = i - 1$
12: until $i < 0$
13: return $x_{0,1}$
```

(due to smoothing of the high frequency errors) but then converges slowly to the solution (due to smoothing of the low frequency errors).

One may observe that low frequency errors appear as higher frequencies on coarser grids. The latter may be effectively reduced using a smoother in the coarse grid. Moreover, smoothing at a coarser level has typically a much lower cost than on finer ones. The core idea of multigrid is to use a sequence of subproblems defined on coarser grids as a means to accelerate the solution process (by relaxation such as Gauss-Seidel) on the finest grid. This leads to recursive algorithms like the so-called V- or W-cycle, which traverse between fine and coarse grids in the mesh hierarchy. Since ultimately only a small number of relaxation steps on each level must be performed, multigrid provides an asymptotically optimal method whose complexity is only $O(N)$, where N is the number of mesh points.

4.2.1. *Multigrid for linear systems.* We recall first the basic of multigrid techniques. We consider a sparse linear system which typically results from the discretization of a partial differential equation on a fine grid Ω_i with a given grid spacing h.

$$A_i x_i = b_i. \tag{4.1}$$

Let $x_{i,0}$ be an initial approximation of (4.1) and x_i be the exact solution. The first step of the algorithm consists in smoothing the error $x_i - x_{i,0}$ by applying N_0 iterations of a relaxation scheme S to (4.1) that has the smoothing property [23]. Examples of the smoother S are Richardson, Jacobi or Gauss-Seidel. The obtained smooth approximation $\bar{x}_{i,1}$ satisfies equation (4.1) up to a residual r_i:

$$A_i x_{i,1} = b_i - r_i.$$

The corresponding error equation is therefore given by

$$A_i e_i = r_i, \tag{4.2}$$

where $e_i = x_{i,1} - x_i$ is the unknown smooth error. Equation (4.2) is then solved on a coarser grid Ω_{i+1} with a grid spacing that has to be larger than h (typical choice is $2h$ but other choices are possible). For this, the fine grid error equation must be approximated by a coarse grid equation:

$$A_{i+1} e_{i+1} = r_{i+1}. \tag{4.3}$$

We need therefore to transfer the residual to the coarse grid and construct a coarse version of the fine matrix. Let R be a *restriction* operator mapping functions on Ω_i to functions on Ω_{i+1} (common examples are injection and full weighting, see [8]). The coarse residual is given then by

$$r_{i+1} = Rr_i.$$

The coarse grid matrix may be obtained by re-discretization on Ω_{i+1} or by using the Galerkin coarse grid approximation: $A_{i+1} = R\, A_i\, P$. Here P is a *prolongation (or interpolation)* operator mapping functions from Ω_{i+1} to Ω_i (standard examples are linear and bilinear interpolation).

The result $e_{i+1,*}$ due to solving (4.3) is transferred back to the fine grid to obtain:

$$e_{i,*} = Pe_{i+1,*}.$$

With this approximation of the fine grid error, the approximate solution is corrected to obtain:

$$x_{i,2} = x_{i,1} + e_{i,*}.$$

In order to damp high frequency error components that might arise due to the interpolation, N_1 iterations of the smoother S are applied on the fine grid to get the new iterate $x_{i,3}$.

The coarse grid problem (4.3) has a lower dimension than the original problem (4.1), but it must be solved accurately for each iteration which can be very costly. In a typical multigrid with three levels or more, this problem is solved by calling recursively the algorithm γ times until reaching the coarsest grid, where we solve (4.3) exactly with a negligible cost. The steps of this multigrid algorithm are summarized in Algorithm 4.

The resulting $x_{i,3}$ may be injected iteratively as initialization in Algorithm 4 until the residual on the finest grid

$$r_{i,3} = b_i - A_i x_{i,3}$$

is smaller than a given tolerance.

Note that for the smoothing steps N_0 and N_1, only the sum is important in a convergence analysis. Typical values for the tuple (N_0, N_1) are $(1,1), (2,1)$ and $(2,2)$. For the cycling parameter γ, only the values 1 and 2 are commonly used.

Algorithm 4 Linear Multigrid V-Cycle

```
1: function MG-cycle($i, A_i, b_i, x_{i,0}$)
2:     $i$: level
3:     $A_i$, $b_i$: defines linear system to be solved
4:     $x_{i,0}$: initialization
5: if i is the coarsest level then
6:     Solve $x_{i,1} = A_i^{-1} b_i$ // Solve exactly
7: else
8:     $x_{i,1} := S(x_{i,0}, A_i, b_i, N_0)$ // Smooth with $N_0$ iterations
9:     $x_{i+1,0} = 0$
10:    $r_i := b_i - A_i x_{i,1}$ // Residual
11:    $r_{i+1} := R\, r_i$ // Restrict
12:    Call MG($i+1, A_{i+1}, r_{i+1}, x_{i+1,0}$) $\gamma$ times // Recursive call
13:    Let $e_{i+1,*}$ be the returned value
14:    $e_{i,*} := P\, e_{i+1,*}$ // Extend
15:    $x_{i,2} := x_{i,1} + e_{i,*}$ // Apply correction step
16:    $x_{i,3} := S(x_{i,2}, A_i, b_i, N_1)$ // Postsmooth with $N_1$ iterations
17:    Return $x_{i,3}$
18: end if
```

The so-called V-cycle corresponds to $\gamma = 1$ and W-cycle corresponds to $\gamma = 2$. This is illustrated in FIGURE 4.1 for three levels.

An important variation of multigrids, which is known as the *full multigrid method* (FMG) [7] or *nested iteration technique* [23], combines a multiresolution approach with a standard multigrid cycle. The FMG starts at the coarsest grid level, solves a very low-dimensional problem, extends the solution to a finer space, performs a multigrid cycle, and repeats the process until a multigrid cycle is performed on the finest grid level. In this way, a good initialization is obtained to start the multigrid cycle on the finest level, which usually reduces the total number of iterations required. The method is illustrated in FIGURE 4.1 for three levels and using a V-cycle.

4.2.2. *Multigrid for optimization problems.* As commented previously, the multigrid strategy has been recently extended to optimization problems for both line search methods and trust region problems. Previous works on non-linear multigrid methods applied the techniques directly to systems of non-linear equations obtained by solving the first-order optimality conditions. This approach is not suitable for problems that are not easily transformed into systems of non-linear equations. Another possible way to use multigrid for optimization problems is to apply the linear multigrid as inner iterations for solving the linear system (3.2) for a given iterate w_k. This idea implicitly assumes that Newton method is the underlying optimization algorithm and that the Hessian matrices can be explicitly computed. Since many large-scale optimization algorithms only require the

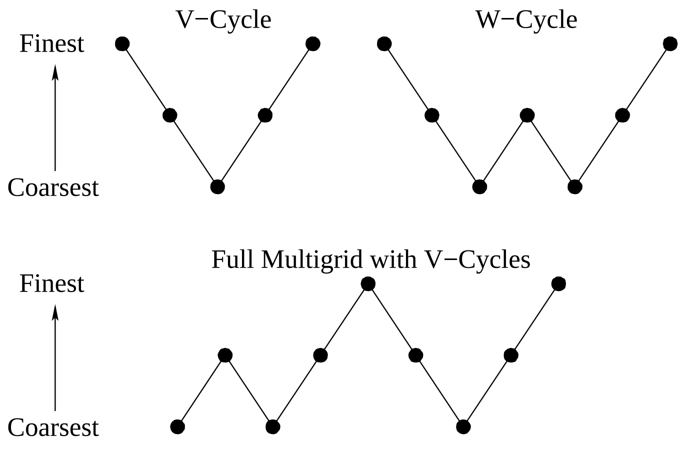

FIGURE 4.1. V-cycle, W-cycle and FMG with V-cycle

computation of function and gradient values, it is less obvious how such multigrid algorithm can be applied. Moreover, solving the Newton equations using multigrid may only lead to a better unilevel optimization algorithm as long as the multigrid technique is not applied in the outer iterations.

We deal here with a multigrid algorithm that works directly with the optimization problem enabling us to work with problems that are not easily transformed into systems of non-linear equations. The multigrid line search optimization (MG/OPT) strategy is described in the following.

As with multigrid for linear systems, in order to solve the optimization problem (2.7) over an original (finest) grid level $i = 0$, a sequence of optimization subproblems are considered on nested coarser grids. Given a fine grid level $i \geq 0$, let f_i denote a representation of the objective function f on this level. Let $w_{i,0}$ be an initial fine approximation to the optimization problem at level i. For the finest level $i = 0$, the optimization problem corresponds to the minimization of f_0, the finest representation of the objective function f. However, for a coarser level i, the optimization problem corresponds to the minimization of a function h_i that shall be specified later. The first step in the multigrid procedure is called a pre-optimization phase (by analogy to pre-smoothing in linear multigrid) and consists in applying N_0 iterations of an optimization procedure like truncated Newton (in our case line search TN) to h_i to obtain w_{i,N_0}. As for nonlinear multigrid, this w_{i,N_0} is transferred to a coarser grid to obtain $w_{i+1,0} := R w_{i,N_0}$. The residual at this level is given by

$$r_{i+1} := \nabla f_{i+1}(w_{i+1,0}) - R \nabla h_i(w_{i,N_0}).$$

The function

$$h_{i+1}(w_{i+1}) = f_{i+1}(w_{i+1}) - r_{i+1}^T w_{i+1} \tag{4.4}$$

is the function to be minimized on the coarse grid level $i+1$. We take $h_0 := f_0 = f$ on the finest level.

Assume that $w_{i+1,*}$ is a solution to the optimization of (4.4). The error between $w_{i+1,*}$ and the initial approximation $w_{i+1,0}$ is called coarse grid correction. This correction is extended back to level i,

$$s_{i,N_0} = P(w_{i+1,*} - w_{i+1,0}).$$

In an optimization context, this correction step used to update the current solution w_{i,N_0} to w_{i,N_0+1} is considered as a search direction called *recursive* to distinguish it from the *direct* search direction step that is computed by a given optimization procedure on the same grid level.

Finally, in order to remove the oscillatory components that may have been introduced by the correction step, one may finish with a post-optimization phase by applying N_1 iterations of the optimization procedure (in our work line search TN) to h_i with initial guess w_{i,N_0+1}, obtaining w_{i,N_0+N_1+1}.

The coarse optimization subproblem for h_{i+1} can be seen as a first order approximation to the fine grid problem h_i since their gradient coincide on $w_{i+1,0}$:

$$\nabla h_{i+1}(w_{i+1,0}) = \nabla f_{i+1}(w_{i+1,0}) - r_{i+1} = R \nabla h_i(w_{i,N_0}).$$

The first-order approximation suggests that the correction will be only reliable near the restricted approximation and it relates at the same time the multigrid optimization algorithm to the steepest descent method. The latter connection indicates that the coarse grid correction will not be typically a well-scaled descent direction which, in turn, implies that a line search should be performed to adjust the scale of the recursive search direction. However, it can be demonstrated that the multigrid algorithm is related to Newton's method, in the sense that the recursive search direction s_i is an approximate Newton direction [29]. Accordingly, in order to improve computational efficiency, the line search for a recursive direction step s_i is performed only if $w_{i,k} + \alpha_k s_i$ with $\alpha_k = 1$ does not reduce the value of h_i. That is, if $h_i(w_{i,k} + s_i) < h_i(w_{i,k})$ we update with $w_{i,k+1} = w_{i,k} + s_i$. Otherwise a line search is performed.

In [29] bound constraints are proposed for the optimization subproblem. In the context of our work, we have seen that bound contraints may improve robustness of the MG/OPT algorithm. These bound constraints may be implemented by means of active sets [39]. However, in our case we have seen that we do not need set up such bounds since the line search TN algorithm used to optimize the subproblem h_{i+1} already provides with similar constraints. Line search algorithms does restrict the search at each iteration $w_{i+1,k}$ to an upper bound which depends on the gradient norm values and thus ensure that the update $w_{i+1,k+1}$ is not far away from $w_{i+1,k}$.

We have implemented the MG/OPT algorithm using a full multigrid method (FMG) to solve the problem and the resulting algorithm will be denoted by FMG/OPT. As for the linear case, the FMG/OPT starts at the coarsest grid level where enough TN iterations are performed, and prolongates the solution to the next finer level i where V_i iterations of the MG/OPT cycle are performed.

Algorithm 5 shows the V-cycle algorithm used for FMG/OPT. At each iteration the algorithm computes a step s_i either directly using the inner iteration of the TN method (Algorithm 2) on the current level, or recursively by means of the multigrid strategy. However, as noted in [47, 22], the recursive call is useful only if $||Rg_{i,k}||$ is large enough compared to $||g_{i,k}||$, where $g_{i,k} = h_i(w_{i,k})$ and $k \leq N_0$. Thus we restrict the use of a coarser level $i+1$ to the case where

$$||Rg_{i,k}|| > \kappa_g ||g_{i,k}|| \text{ and } ||Rg_{i,k}|| > \epsilon_g \tag{4.5}$$

for some constant $\kappa_g \in (0, \min ||R||)$ and where $\epsilon_g \in (0,1)$ is a measure of first order criticality for h_{i+1}. The latter condition is easy to check before trying to compute a step at level $i+1$.

5. Implementation issues

The numerical algorithms line search TN, MR/OPT and FMG/OPT have been implemented in C using MegaWave2 library. In this section, we provide details about derivatives computation for both the objective function and the image.

5.1. **Functional gradient calculation.**

The gradient of the objective function in (2.7) is calculated analytically and given by

$$g = \nabla f = \begin{pmatrix} f^u \\ f^v \end{pmatrix} = \begin{pmatrix} D^u + \alpha R^u \\ D^v + \alpha R^v \end{pmatrix}.$$

5.1.1. *Horn-Schunck data term.* For the linear data term we have

$$D(u,v) = \frac{1}{2} \sum_{i,j} \psi \left(\left[I^x u_{i,j} + I^y v_{i,j} + I^t \right]^2 \right).$$

where i (respectively j) corresponds to the discrete column (respectively row) of the image, being the coordinate origin located in the top-left corner of the image. The function ψ is used to enhance robustness with respect to outliers. In our work we have used

$$\psi\left(x^2\right) = \begin{cases} x^2 & \text{if } |x| \leq \gamma \\ \gamma^2 & \text{otherwise,} \end{cases}$$

where γ is a given threshold. The gradient D for $|x| \leq \gamma$ is therefore given by

$$\begin{pmatrix} D^u_{i,j} \\ D^v_{i,j} \end{pmatrix} = \begin{pmatrix} I^x \left[I^x u_{i,j} + I^y v_{i,j} + I^t \right] \\ I^y \left[I^x u_{i,j} + I^y v_{i,j} + I^t \right] \end{pmatrix},$$

where $D^u_{i,j}$ and $D^v_{i,j}$ refer to the partial derivative of $D(u,v)$ with respect to variables $u_{i,j}$ and $v_{i,j}$, respectively. Here I^x, I^y, I^t are the spatial and temporal image derivatives for which the computation is explained in Section 5.3. Note that for $|x| > \gamma$ the gradient D is $(D^u_{i,j}, D^v_{i,j})^T = (0,0)^T$.

Algorithm 5 The V-cycle for the FMG/OPT algorithm

```
1: function MG/OPT-cycle($i$, $h_i$, $w_{i,0}$)
2:     i: level
3:     $h_i$: coarse objective function at level $i$ // $h_0 = f$
4:     $w_{i,0}$: initial approximation to $h_i$
5: if i is coarsest level then
6:     task = optimize, $N$ = max_outer
7: else
8:     task = pre-optimize, $N = N_0$
9: end if
10: $k_0 = k = 0$
11: loop
12:     if task is optimize, pre-optimize or post-optimize then
13:         Compute $s_{i,k}$ by calling Algorithm 2
14:     else
15:         $w_{i+1,0} = Rw_{i,k}$ // Restrict
16:         $r_{i+1} = \nabla f_{i+1}(w_{i+1,0}) - R\nabla h_i(w_{i,k})$ // Residual
17:         $h_{i+1}(w_{i+1}) = f_{i+1}(w_{i+1}) - r_{i+1}^T w_{i+1}$ // Next coarse objective function
18:         Call MG/OPT-cycle(i+1, $h_{i+1}$, $w_{i+1,0}$)
19:         Let $w_{i+1,*}$ be the returned solution
20:         $s_{i,k} = P(w_{i+1,*} - w_{i+1,0})$ // Correction step
21:     end if
22:     Perform a line search to scale the step $s_{i,k}$ by $\alpha_k$
23:     $w_{i,k+1} = w_{i,k} + \alpha_{i,k} s_{i,k}$
24:     Update $M_{i,k+1}$ by the BFGS formula
25:     if $|f_{i,k+1} - f_{i,k}| < \epsilon_f$ or $||w_{i,k+1} - w_{i,k}|| < \epsilon_x$ then
26:         return $w_{i,k+1}$
27:     end if
28:     $k := k + 1$
29:     // Select next task to do
30:     if task is pre-optimize and (4.5) is satisfied then
31:         task = recursive-call
32:     else if task is recursive-call then
33:         task = post-optimize, $N = N_1$, $k_0 = k$
34:     end if
35:     // Check if maximum number of outer iterations has been reached
36:     if (task is optimize, pre-optimize or post-optimize) and ($k - k_0 = N$) then
37:         return $w_{i,k}$
38:     end if
39: end loop
```

5.1.2. *Intensity constancy based data term.* The nonlinear data term based on the constancy assumption is as follows

$$D(u,v) = \frac{1}{2}\sum_{i,j} \psi\left([I_2(i+u_{i,j}, j+v_{i,j}) - I_1(i,j)]^2\right).$$

The gradient of this functional for $|x| \le \gamma$ is given by

$$\begin{pmatrix} D^u_{i,j} \\ D^v_{i,j} \end{pmatrix} = \begin{pmatrix} I_2^x(i+u_{i,j}, j+v_{i,j})\,[I_2(i+u_{i,j}, j+v_{i,j}) - I_1(i,j)] \\ I_2^y(i+u_{i,j}, j+v_{i,j})\,[I_2(i+u_{i,j}, j+v_{i,j}) - I_1(i,j)] \end{pmatrix}.$$

5.1.3. *Quadratic regularization term.* Now we consider the regularization term and start by the quadratic functional. We have

$$R(u,v) = \frac{1}{2}\sum_{i,j} ||\nabla_{i,j}(u,v)||^2,$$

where $||\nabla_{i,j}(u,v)|| = \sqrt{(u^x_{i,j})^2 + (u^y_{i,j})^2 + (v^x_{i,j})^2 + (v^y_{i,j})^2}$. The partial derivatives of u, v are computed by forward finite differences with a discretization step h, that is, $u^x_{i,j} = h^{-1}(u_{i+1,j} - u_{i,j})$ and $u^y_{i,j} = h^{-1}(u_{i,j+1} - u_{i,j})$ (derivatives v^x and v^y are computed similarly). The gradient of this functional is obtained as

$$\begin{pmatrix} R^u_{i,j} \\ R^v_{i,j} \end{pmatrix} = \frac{1}{h^2}\begin{pmatrix} 4u_{i,j} - u_{i-1,j} - u_{i,j-1} - u_{i+1,j} - u_{i,j+1} \\ 4v_{i,j} - v_{i-1,j} - v_{i,j-1} - v_{i+1,j} - v_{i,j+1} \end{pmatrix}.$$

5.1.4. *Total variation term.* In this case we suppose that

$$R(u,v) = \sum_{i,j} ||\nabla_{i,j}(u,v)||.$$

To overcome the problem of non-differentiability of the total variation, a widely spread technique consists in approximating R by a differentiable function:

$$R(u,v) \approx \sum_{i,j} \psi_{i,j},$$

where $\psi_{i,j} = \sqrt{(u^x_{i,j})^2 + (u^y_{i,j})^2 + (v^x_{i,j})^2 + (v^y_{i,j})^2 + \mu}$ and μ is a small positive parameter. Using again forward finite differences, the gradient of the last approximation is given by

$$\begin{pmatrix} R^u_{i,j} \\ R^v_{i,j} \end{pmatrix} = \frac{1}{h}\begin{pmatrix} \frac{u^x_{i-1,j}}{\psi_{i-1,j}} + \frac{u^y_{i,j-1}}{\psi_{i,j-1}} - \frac{u^x_{i,j}+u^y_{i,j}}{\psi_{i,j}} \\ \frac{v^x_{i-1,j}}{\psi_{i-1,j}} + \frac{v^y_{i,j-1}}{\psi_{i,j-1}} - \frac{v^x_{i,j}+v^y_{i,j}}{\psi_{i,j}} \end{pmatrix}.$$

Another approximation of the total variation is given by

$$R(u,v) \approx \sum_{i,j} \varphi_\mu\left(||\nabla_{i,j}(u,v)||\right) \quad \text{where} \quad \varphi_\mu(x) = \begin{cases} |x| & \text{if } |x| \ge \mu \\ x^2/2\mu + \mu/2 & \text{otherwise.} \end{cases}$$

Theoretically, this approximation is twice better than the first standard approximation. However, numerically in our implementaion, both approximations lead to the same results.

5.2. **Hessian calculation.** For computing the Newton direction in the tuncated Newton method, the linear conjugate gradient is a Hessian-free procedure (see Algorithm 2) and needs only to supply a routine for computing the product of the Hessian with Newton direction p. This matrix-vector product is computed via forward finite differences:

$$H(w)\,p = \frac{g(w+\epsilon p) - g(w)}{\epsilon}$$

where ϵ is chosen to be the square root of the machine precision divided by the norm of w.

5.3. **Image gradient calculation.** Differentiation is an ill-posed problem [5] and regularization may be used to obtain good numerical derivatives. Such regularization may be accomplished with a low-pass filter such as the Gaussian, and is essential for motion estimation [4, 43]. More recently, [19] proposes to use a matched pair of low pass and differentiation filters as a gradient operator which are the ones used in this work.

Usually, the derivative at an integer $I^x_{i,j}$ is computed by applying a separable filter composed by a Gaussian filter (alternatively, a matched low pass filter) in the y direction and the derivative of the Gaussian (alternatively, a matched derivative) is applied in the x direction. Conversely, the computation of the derivative in the y direction at an integer point is performed by applying a separable filter composed by a Gaussian (or matched low pass filter) in the x direction and the derivative of the Gaussian (or a matched derivative) in the y direction.

For motion estimation applications it may be necessary to compute the gradient at non integer points, since non integer displacements are allowed. This is the case, for instance, of the nonlinear data term of Section 5.1.2. In such cases, a simple way to proceed is a two step process: in a first step, the original image is interpolated at the required points using a bilinear interpolation or an interpolation kernel such as [28], and then the derivative is computed using the latter interpolated points. Another way to proceed is to first compute the gradient at integer points and then apply an interpolation kernel over the gradient values using these latter values. Both procedures are theoretically equivalent since differentiation and interpolation are based on linear operators (and thus, they are interchangeable). Let us call this approach as linear gradient interpolation.

In this work the computation of the gradient at non-integer points is done by means of a shift in the Fourier domain. Moreover, rather than shifting the image (or gradient) to obtain the interpolated values, the derivative filter taps are shifted so as to obtain the filter coefficients that have to be applied on the

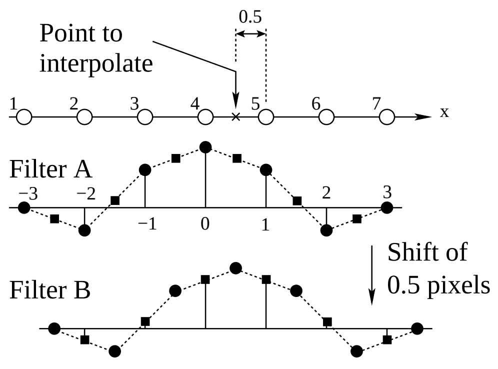

FIGURE 5.1. Filter tap interpolation by means of a shift in the Fourier domain.

integer image values in order to obtain the corresponding interpolated gradient value.

Assume that the gradient and interpolation kernels are linear, shift-invariant and separable. Such kernels may be found in [19]. Without loss of generality, the interpolation problem in the Fourier domain can be thus restricted to one dimension. Let us consider FIGURE 5.1 which shows the proposed technique. On top several samples of a one dimensional signal are shown. Filter taps A are used to obtain the gradient at integer points (in the example the filter A is centered on $x = 4$), whereas filter taps B may be used to obtain the gradient at every non-integer point half-way between two integer points (in the example the filter B is centered on $x = 4.5$). The filter taps B are obtained from filter taps A by means of a shift of 0.5 in the Fourier domain. They can be thus be applied directly on the original data. We will call this procedure Fourier gradient interpolation.

FIGURE 5.2 shows an example in which a set of matched filters [19] of size 9 are interpolated at non-integer points by a shift in the Fourier domain and by a linear interpolation. Performing a linear interpolation on the filter taps in order to apply them to the original data values is equivalent to the gradient linear interpolation approach described above. Note that, as expected, the obtained filter taps are different for both methods. The experimental section will show that Fourier based interpolation leads to a better performance than linear interpolation, especially in the multigrid approach.

In the two-dimensional case, if I^x has to be computed at point $(i+\Delta i, j+\Delta j)$, where (i, j) is an integer discrete image position and $\Delta i < 1$ and $\Delta j < 1$, the matched low-pass filter in the y direction (respectively the matched derivative in the x direction) is obtained by shifting the corresponding taps Δi (respectively Δj) in the Fourier domain. A similar procedure is used to compute I^y at a non-integer point.

The previous scheme has a high computational load if the Fourier shift has to be applied to each non-integer position where one needs to interpolate the gradient.

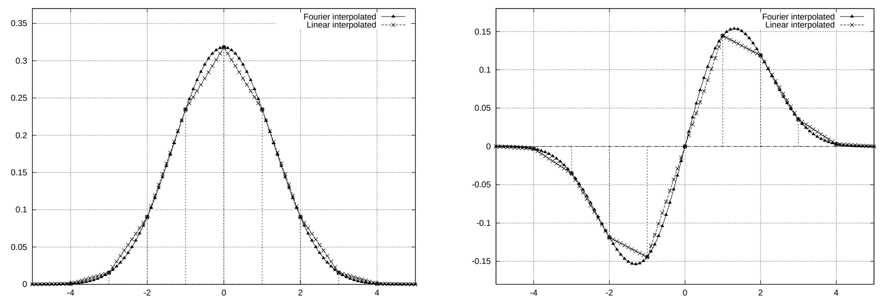

FIGURE 5.2. A set of matched filters of size 9 are interpolated at non-integer points by a shift in the Fourier domain (black) and by a linear interpolation (light gray).

The computational load can be reduced significantly if, at the initialization of the optimization algorithm, the gradient is computed with the Fourier gradient interpolation at all points of a grid of size h/D. Then, each time one has to compute the gradient at a non-integer point a bilinear interpolation with the neighboring pixels is used. The FIGURE 5.2 shows the case where $D = 10$.

6. EXPERIMENTAL RESULTS

To assess the performance of the proposed algorithms, we use three classical sequences of synthetic images that consist of scenes of various complexity; namely, the translating tree, the diverging tree and the Yosemite sequences. The reference frame and the corresponding ground truth of the synthetic sequences are show in Figures 6.1 and 6.2. The image size of the tree sequences is 150×150 while the Yosemite sequence is of size 316×252.

Since we are more interested in the computational complexity of the proposed algorithms, we compare first the CPU time needed by each numerical algorithm to reach a similar accuracy using four optical flow models. All tests are performed on a PC with a 2.0 GHz Mobile Intel DualCore processor and 1 GB RAM. We compute also the number of functional and gradient evaluations that were performed by each algorithm to reach the estimated optical flow. We measure the overall number of function evaluations as

$$N_f = \sum_{i=0}^{L-1} \frac{N_{f,i}}{F_i}$$

where $N_{f,i}$ is the number of function evaluations performed by the optimization algorithm at resolution level i. For the TN and our MR/OPT and FMG/OPT algorithms, the function is evaluated during the line search procedure. F_i is the mesh resolution ratio of a given level i with respect to the finest level 0. In this work $F_i = 2^{2i}$.

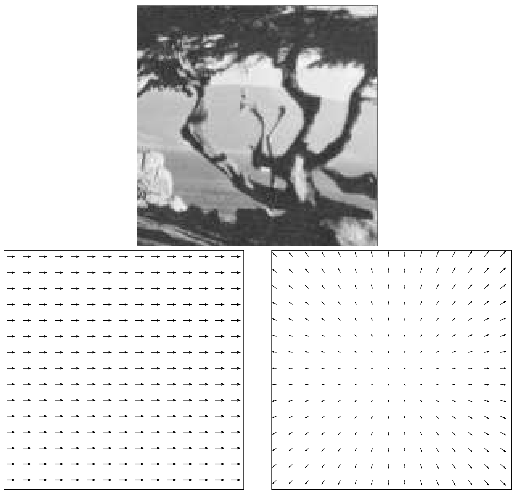

Figure 6.1. On the top one frame of the original sequence is shown. On the bottom the ground truth for the corresponding translating (left) and diverging (right) is shown. Motion vectors have been scaled by a factor of 2.5 for better visibility.

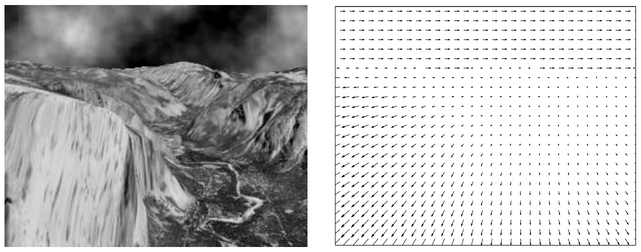

Figure 6.2. On the left the one frame of the Yosemite sequence is shown. On the right the corresponding ground truth is depicted where motion vectors have been scaled by a factor of 2.5 for better visibility.

The number of gradient evaluations is defined in a similar manner. While the function is only evaluated during the line search procedure, the gradient is additionally evaluated during the inner iterations of the TN algorithm (i.e. the Hessian computation). Thus, we expect the number of function evaluations to be always lower than the gradient evaluations. Note also that the number of gradient evaluations is more crucial to the overall computational work than that of the objective function. Here one gradient evaluation is approximately equivalent to two function evaluations when using quadratic regularization, while it takes almost three times in the case of TV regularization.

Moreover, we compare the quality of the optical flow estimations of the numerical algorithms. For this, we measure the average angular error (AAE) and the standard deviation (STD) of the estimated flow w^e with respect to the ground truth w^c. For a given pixel (i, j), the angular error (AE) between the ground truth motion vector, $w^c_{i,j}$, and the estimated flow, $w^e_{i,j}$, is computed as:

$$\mathrm{AE}(w^c_{i,j}, w^e_{i,j}) = \cos^{-1}\left(\frac{u^c_{i,j}u^e_{i,j} + v^c_{i,j}v^e_{i,j} + 1}{\sqrt{(u^c_{i,j})^2 + (v^c_{i,j})^2 + 1}\sqrt{(u^e_{i,j})^2 + (v^e_{i,j})^2 + 1}}\right).$$

The average angular error is the mean of the angular error over all pixels N_{np} of the image

$$\mathrm{AAE}(w_c, w_e) = \frac{1}{N_{np}}\sum_{i,j}\mathrm{AE}(w^c_{i,j}, w^e_{i,j}).$$

The standard deviation is computed as

$$\mathrm{STD}(w_c, w_e) = \sqrt{\frac{1}{N_{np}}\sum_{i,j}\left(\mathrm{AE}(w^c_{i,j}, w^e_{i,j}) - AAE(w_c, w_e)\right)^2}.$$

Before going through the performance evaluation of multilevel optimization algorithms, we first demonstrate the competitiveness of the adopted discretize-optimize approach (numerical optimization) versus the standard optimize-discretize approach (Gauss-Seidel) and also justify the selection of the chosen numerical optimization algorithm. To this end, we compare the CPU time needed to reach a similar AAE by the following algorithms: 1) the proposed line-search two-step preconditionned truncated Newton algorithm approach (see Section 3), called here TN1, 2) the line-search Quasi-Newton L-BFGS approach of [30], called here QN. A positive-definite approximation to the Hessian is obtained at each iteration w_k by storing the previous steps w_k and the BFGS approach. Instead of solving at each outer iteration the Newton equation, the L-BFGS method takes advantge from the fact that the obtained BFGS preconditionning matrix is easily invertible. Thus, in the L-BFGS approach the line 7 of Algorithm 1 is substituted by $s_k = M_k^{-1}g_k$. The proposed algorithm TN1 is also compared against 3) a line-search L-BFGS precondionned truncated Newton method, called here TN2. This corresponds to the same algorithm as TN1 but the L-BFGS preconditionning approach [30] is used instead the two-step BFGS approach [37].
It should be noted that in these three numerical optimization algorithms (TN1, TN2 and QN) the same line-search approach is used [34]. The previous algorithms were compared using the linear data term and quatratic regularization, which corresponds to the classical Horn-Schunck model. Thus we implemented also a Gauss-Seidel Horn-Schunck smoother [25]. We call the latter algorithm GS.
The experimental results for the four algorithms (GS, TN1, TN2 and QN) are shown in Table 6.1. Best results for QN were obtained using 4 steps, and TN2 was setup to the same number of steps for the preconditioner. As expected from a

Table 6.1. CPU time needed by the approaches GS, TN1, TN2 and QN to reach similar accuracy for the Yosemite sequence using the Horn-Schunck model.

Algorithm		GS	TN1	TN2	QN
CPU Time (s)	Unilevel	45.22	8.90	9.36	17.20
	Multiresolution	10.74	5.21	5.65	9.50
AAE		7.40	6.81	6.82	6.85

linear relaxation scheme, Gauss-Seidel has problems near the solution and indeed in the tests it can reach an AAE below 8.00 in less than 1 second but will get stuck and does not reach the optimal solution. Overall, truncated Newton algorithms perform better justifying the choice of the TN as the underlying optimization smoother within multilevel algorithms to determine direct search directions. TN1 and TN2 perform almost the same with a slight better performance of TN1. This algorithm will be used in the sequel as the smoother and will be denoted by OPT.

Now we report the performance evaluation of the unilevel, the multiresolution and the multigrid optimization algorithms applied to estimate the optical flow between two successive frames of the above three sequences. We have considered four optical flow models as described above in Section 2 and Subsection 5.1. We note that we use thresholding in all the data terms to remove outliers. In all experiments we have considered $L = 6$ levels of resolution for multilevel algorithms (multiresolution and multigrid). The stopping criteria for all algorithms were set on the relative error of the objective function, the gradient norm or the solution norm with a tolerance of 10^{-5}. For the MR/OPT algorithm, the maximum number of outer iterations was set to 10 iterations per each resolution for all the optical flow models except when using the nonlinear data term and the TV regularization for which we use a maximum of 15 iterations. For the FMG/OPT, we perform 2 or 3 V-cycles when using the nonlinear data term, while only 1 V-cycle is sufficient in the case of the linear data term. We note here that for the purpose of a fair computational work comparison, we stop the outer iterations once a similar accuracy on the optical flow estimation is reached. For all the algorithms, the maximum number of innner iterations within the OPT method was set to 20.

In Tables 6.2-6.4, we summarize the quality of the solution and the computational costs of the three numerical algorithms for four optical flow models. Model 1 refers to the linear data term plus the quadratic regularization; model 2 refers to the nonlinear data term plus the quadratic regularization; model 3 refers to the linear data term plus the TV regularization; and finally model 4 refers to the nonlinear data term plus the TV regularization. In terms of the quality of the solution, by comparing the unilevel algorithm versus the multilevel algorithms, we note that the optical flow estimation of the four models is more accurate when

Table 6.2. Comparison of computational work and optical flow Estimation for two Frames of the translating tree sequence using optimization algorithms OPT, MR/OPT and FMG/OPT. Time is CPU time in seconds.

		OPT	MR/OPT	FMG/OPT
Model 1	Time	11.91	2.73	0.98
	Nf	69	15	12
	Ng	572	133	37
	AAE	1.83	0.90	0.89
	STD	1.23	0.55	0.51
Model 2	Time	14.22	5.40	2.07
	Nf	89	19	25
	Ng	586	221	72
	AAE	0.27	0.23	0.23
	STD	0.29	0.22	0.22
Model 3	Time	27.78	4.90	2.40
	Nf	95	15	21
	Ng	1042	189	87
	AAE	1.54	0.77	0.75
	STD	0.86	0.47	0.42
Model 4	Time	35.32	10.25	4.08
	Nf	125	35	46
	Ng	1130	334	126
	AAE	0.20	0.20	0.20
	STD	0.18	0.18	0.17

computed using the latter algorithms for all the tested images. In this regard, multigrid optimization has shown to provide a more accurate estimation than multiresolution optimization if we take the average angular error as an accuracy measure, see Tables 6.2-6.4.

In overall, the FMG/OPT algorithm performs at least twice better than the MR/OPT algorithm and ten times better than the unilevel truncated Newton algorithm; see Table 6.5. We notice also that the FMG/OPT algorithm is less independent of the image size because it often takes similar number of function and gradient evaluations while comparing across the same optical flow model.

7. Conclusion

Based on the discretize-optimize approach, we have applied different numerical optimization techniques to variational models for optical flow computation. First, we have shown the competitiveness of this strategy compared to the classical optimize-discretize approach. Three Newton-based optimization algorithms were superior to the Gauss-Seidel method when applied to the classical Horn-Schunck

TABLE 6.3. Comparison of computational work and optical flow estimation for two frames of the Diverging Tree sequence using optimization algorithms OPT, MR/OPT and FMG/OPT. Time is CPU time in seconds.

		OPT	MR/OPT	FMG/OPT
	Time	14.56	4.10	0.72
	Nf	64	16	10
Model 1	Ng	695	209	27
	AAE	2.06	1.99	1.82
	STD	1.60	1.57	1.30
	Time	16.81	5.28	1.70
	Nf	82	20	16
Model 2	Ng	664	221	63
	AAE	2.08	2.07	1.90
	STD	1.95	2.09	1.85
	Time	26.59	6.61	1.06
	Nf	128	20	11
Model 3	Ng	1016	253	35
	AAE	2.49	2.46	1.59
	STD	1.86	1.95	1.17
	Time	31.60	14.73	3.26
	Nf	133	96	30
Model 4	Ng	997	492	97
	AAE	2.09	2.07	1.86
	STD	1.84	1.89	1.92

model. In particular, truncated Newton was shown to be a suitable unilevel optimization algorithm and was chosen as the smoother for optimization-based multilevel methods. We have then implemented the FMG/OPT algorithm based on a line search strategy to scale the (direct) Newton or the (recursive) multigrid search direction. Several components of the MG/OPT technique have been tuned for high efficiency and the algorithm has been fully evaluated with respect to unilevel and (one-way) multiresolution optimization. Our experimental results have demonstrated that the FMG/OPT algorithm can be effectively used for optical flow computation. Using different models and images, we have observed the FMG/OPT algorithm was faster and more accurate than both unilevel and multiresolution truncated Newton. Further research will investigate the use of line search multigrid versus trust region multigrid in the context of dense optical flow computation. The proposed numerical strategy can be adapted to the minimization of other nonlinear energy functionals like the illumination invariant model proposed in [14] or depth estimation in stereo problems.

Table 6.4. Comparison of computational work and optical flow estimation for two frames of the Yosemite sequence using line search optimization algorithms (OPT, MR/OPT, FMG/OPT). Time is CPU time in seconds.

		OPT	MR/OPT	FMG/OPT
Model 1	Time	12.07	5.21	2.97
	Nf	13	8	11
	Ng	161	60	26
	AAE	6.79	6.78	6.77
	STD	9.25	9.31	9.49
Model 2	Time	62.20	15.59	7.06
	Nf	108	27	19
	Ng	680	167	67
	AAE	6.47	6.22	6.17
	STD	9.28	9.15	9.18
Model 3	Time	82.27	17.38	7.01
	Nf	83	19	15
	Ng	856	188	81
	AAE	6.33	6.07	6.08
	STD	9.10	8.62	8.40
Model 4	Time	205.56	25.51	14.46
	Nf	218	24	21
	Ng	1821	223	121
	AAE	5.75	5.49	5.45
	STD	8.92	8.57	7.90

Table 6.5. Global characteristics of OPT, MR/OPT and FMG/OPT for optical flow models on all the three images. Nfg is the total number of function and gradient evaluations.

		OPT	MR/OPT	FMG/OPT
Model 1	Total time (s)	38.54	12.04	4.76
	Total Nfg	1501	421	106
Model 2	Total time (s)	93.63	26.27	10.83
	Total Nfg	2069	642	232
Model 3	Total time (s)	136.64	28.89	10.47
	Total Nfg	3016	648	219
Model 4	Total time (s)	272.48	50.49	21.80
	Total Nfg	3117	1101	376
All models	**Total time (s)**	**541.29**	**117.69**	**47.86**
	Total Nfg	**9703**	**2812**	**933**

References

[1] L. Álvarez, J. Weickert, and J. Sánchez, *A scale-space approach to nonlocal optical flow calculations*, in Scale-Space Theories in Computer Vision, vol. 1682 of Lecture Notes in Computer Science, Springer, 1999, pp. 235–246.

[2] ——, *Reliable estimation of dense optical flow fields with large displacements*, International Journal of Computer Vision, 39 (2000), pp. 41–56.

[3] P. Anandan, *A computational framework and an algorithm for the measurement of visual motion*, International Journal of Computer Vision, 2 (1989), pp. 283–310.

[4] J. Barron, D. Fleet, and S. Beauchemin, *Performance of optical flow techniques*, International Journal of Computer Vision, 12 (1994), pp. 43–77.

[5] M. Bertero, T. A. Poggio, and V. Torre, *Ill-posed problems in early vision*, Proceedings of the IEEE, 76 (1988), pp. 869–889.

[6] A. Borzì, K. Ito, and K. Kunisch, *Optimal control formulation for determining optical flow*, SIAM Journal on Scientific Computing, 24 (2002), pp. 818–847.

[7] A. Brandt, *Multi-level adaptive solutions to boundary-value problems*, Mathematics of Computation, 31 (1977), pp. 333–390.

[8] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial, 2nd ed.*, SIAM Press, Philadelphia, 2000.

[9] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, *High accuracy optical flow estimation based on a theory for warping*, in European Conference in Computer Vision, vol. 4, 2004, pp. 25–36.

[10] A. Bruhn, *Variational optic flow computation: Accurate modeling and efficient numerics*, PhD thesis, Department of Mathematics and Computer Science, Saarland University, 2006.

[11] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnorr, *Real-time optic flow computation with variational methods*, in International Conference on Computer Analysis of Images and Patterns, 2003, pp. 222–229.

[12] ——, *Variational optical flow computation in real time*, IEEE Transactions on Image Processing, 14 (2005), pp. 608–615.

[13] P. Y. Burgi, *Motion estimation based on the direction of intensity gradient*, Image and Vision Computing, 22 (2004), pp. 637–653.

[14] V. Caselles, L. Garrido, and L. Igual, *A contrast invariant approach to motion estimation*, in Scale Space, Springer Lecture Notes in Computer Science, 2005, pp. 242–253.

[15] I. Cohen and I. Herlin, *Non uniform multiresolution method for optical flow and phase portrait models: Environmental applications*, International Journal of Computer Vision, 33 (1999), pp. 1–22.

[16] R. S. Dembo and T. Steihaug, *Truncated Newton algorithms for large scale unconstrained optimization*, Mathematical Programming, 26 (1983), pp. 190–212.

[17] M. Droske and M. Rumpf, *A variational approach to nonrigid morphological image registration*, SIAM Journal of Applied Mathematics, 64 (2004), pp. 668–687.

[18] W. Enkelmann, *Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences*, Computer Vision, Graphics and Image Processing, 43 (1988), pp. 150–177.

[19] H. Farid and E. P. Simoncelli, *Differentiation of discrete multidimensional signals*, IEEE Transactions on Image Processing, 13 (2004), pp. 496–508.

[20] S. Ghosal and P. A. Vanek, *Fast scalable algorithm for discontinuous optical flow estimation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 18 (1996), pp. 181–194.

[21] F. Glazer, *Multilevel relaxation in low-level computer vision*, in Multi-resolution image processing and Analysis, A. Rosenfeld, ed., Springer-Verlag, 1984, pp. 312–330.

[22] S. Gratton and P. L. Toint A. Sartenaer, *Recursive trust region method for multiscale nonlinear optimization*, SIAM Journal on Optimization, 19 (2008), pp. 414–444.

[23] W. Hackbusch, *Multi-Grid Methods and Applications*, Springer, Berlin/New York, 1985.

[24] W. Hinterberger, O. Scherzer, C. Schnörr, and J. Weickert, *Analysis of optical flow models in the framework of the calculus of variations*, Numerical Functional Analysis and Optimization, 23 (2002), pp. 69–89.

[25] B. Horn and B. Schunk, *Determining optical flow*, Artificial Intelligence, 20 (1981).

[26] E. M. Kalmoun, H. Köstler, and U. Rüde, *3d optical flow computation using a parallel variational multigrid scheme with application to cardiac c-arm ct motion*, Image and Vision Computing, 25 (2007), pp. 1482–1494.

[27] E. M. Kalmoun and U. Rüde, *A variational multigrid for computing the optical flow*, in Proceedings of the Vision, Modeling, and Visualization Conference, Akademische Verlagsgesellschaft, 2003, pp. 577–584.

[28] R. G. Keys, *Cubic convolution interpolation for digital image processing*, IEEE Trans. Acoustics, Speech, and Signal Processing, 29 (1981), pp. 1153–1160.

[29] R. M. Lewis and S. G. Nash, *Model problems for the multigrid optimization of systems governed by differential equations*, SIAM Journal on Scientific Computing, 26 (2005), pp. 1811–1837.

[30] D. C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization*, Mathematical Programming, 45 (1989), pp. 503–528.

[31] B. D. Lucas and T. Kanade, *An iterative image registration technique with an application to stereo vision*, in Image Understanding Workshop, 1981, pp. 121–130.

[32] E. Memin and P. Perez, *Dense estimation and object-based segmentation of the optical-flow with robust techniques*, IEEE Transactions on Image Processing, 7 (1998), pp. 703–719.

[33] ——, *A multigrid approach for hierarchical motion estimation*, in Proceedins of the Sixth International Conference on Computer Vision, 1998, pp. 933–938.

[34] J. J. Moré and D. J. Thuente, *Line search algorithms with guaranteed sufficient decrease*, ACM Transactions on Mathematical Software, 20 (1994), pp. 286–307.

[35] H.-H. Nagel, *Constraints for the estimation of displacement vector fields from image sequences*, in Proceedings of the Eight International Joint Conference on Artificial Intelligence, Alan Bundy, ed., Karlsruhe, Germany, Aug. 1983, William Kaufmann, pp. 945–951.

[36] H.-H. Nagel and W. Enkelmann, *An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 8 (1986), pp. 565–593.

[37] S. G. Nash, *Preconditioning of truncated-newton methods*, SIAM Journal on Scientific and Statistical Computing, 6 (1985), pp. 599–616.

[38] ——, *A multigrid approach to discretized optimization problems*, Optimization Methods and Software, 14 (2000), pp. 99–116.

[39] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.

[40] J. M. Odobez and P. Bouthemy, *Robust multiresolution estimation of parametric motion models*, Journal of Visual Communication and Image Representation, 6 (1995), pp. 348–365.

[41] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert, *Highly accurate optic flow computation with theoretically justified warping*, International Journal of Computer Vision, 67 (2006), pp. 141–158.

[42] C. Koch R. Battiti, E. Amaldi, *Computing optical flow across multiple scales: an adaptive coarse-to-fine strategy*, International Journal of Computer Vision, 6 (1991), pp. 133–145.

[43] C. Stiller and J. Konrad, *Estimating motion in image sequences*, IEEE Signal Processing Magazine, 16 (1999), pp. 70–91.

[44] R. Szeliski and H. Y. Shum, *Motion estimation with quadtree splines*, IEEE Trans. Pattern Analysis and Machine Intelligence, 18 (1996), pp. 1199–1210.

[45] D. Terzopoulos, *Image analysis using multigrid relaxation methods*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 8 (1986), pp. 129–139.

[46] J. Weickert and C. Schnorr, *A theoretical framework for convex regularizers in PDE-based computation of image motion*, International Journal of Computer Vision, 45 (2001), pp. 245–264.

[47] Z. Wen and D. Goldfarb D., *A Line Search Multigrid Method for Large-Scale Convex Optimization*, tech. report, Department of IEOR, Columbia University, 2007.

[48] D. Xie and T. Schlick, *Efficient implementation of the truncated-Newton algorithm for large-scale chemistry applications*, SIAM Journal on Optimization, 10 (1999), pp. 132–154.

El Mostafa Kalmoun
Faculty of Quantitative Sciences
College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok, Malaysia
E-mail address: `elmostafa@uum.edu.my`

Luis Garrido
Departament de Matemàtica Aplicada i Anàlisi
Universitat de Barcelona
Barcelona, Spain
E-mail address: `lluis.garrido@ub.edu`

Vicent Caselles
Departament de Tecnologia
Universitat Pompeu Fabra
Barcelona, Spain
E-mail address: `vicent.caselles@upf.edu`